木パッキング問題について(計算アルゴリズムの基礎理論)

AUTHOR(S):

増山, 繁

CITATION:

増山, 繁. 木パッキング問題について(計算アルゴリズムの基礎理論). 数理解析研究所講究録 1987, 625: 137-146

ISSUE DATE:

1987-05

URL:

http://hdl.handle.net/2433/99957

数理解析研究所講究録
625巻 1987年 137-146

木パッキング問題について

(On the Tree Packing Problems)

京都大学工学部　増山　繁

Shigeru MASUYAMA
Faculty of Engineering, Kyoto University

あらまし　本稿では，木を木に最大個数詰める最大木パッキング問題[10]について調べる．まず，根付き有向木に根付き有向木を詰める場合に，最大節点パッキング(節点の重なりも辺の重なりも許さない)を求める$O(n^2)$(nはTの節点数)のアルゴリズム，及び，最大辺パッキング(節点の重なりは許すが辺の重なりは許さない)を求める多項式時間のアルゴリズムを与えた．それに基づき，無向木への無向木の最大節点パッキングを求める$O(n^2)$(以下では，nは節点数)のアルゴリズム，及び，最大辺パッキングを求める多項式時間のアルゴリズムを与えた．また，根付き櫛形木を根付き有向木に詰める場合，k星を無向木に詰める場合など，いくつかの特別の場合に対しては，O(n)のアルゴリズムを得た．

1．はじめに

木パッキング問題は，与えられた木Tに木T_0を互いに節点も辺も重ならないようにして最大個数詰める最大節点パッキング問題と，節点は互いに重なってもよいが辺は重ならないようにして最大個数詰める最大辺パッキング問題からなる．節点パッキング問題と，それに関連したグラフの因子分解問題は，グラフ理論における古典的な問題のひとつであって(たとえば[5]参照)，従来よく研究されており，特に，k鎖節点パッキング問題[9]では，k＝1の場合はグラフ上のマッチング問題[8]となることから，グラフのマッチング問題の一般化とみなせる．グラフのマッチング問題に対しては効率のよい解法が知られている[8]が，k鎖節点パッキング問題は，k≧2のとき，NP完全である[4]．なお，NP完全な問題に対しては多項式時間のアルゴリズムが存在しないと一般に信じられている．更にKirkpatrickとHell[7]により，グラフを節点パッキングする問題においては，節点と辺のみからなる部分グラフを詰める場合，つまり，実質的にマッチングの場合を除いてはNP完全であることも示されている．一方，辺パッキング問題に関しては，文献[12]でグラフの構造を入れて一般化したビンパッキング問題の性質を調べ，また，Jüngerら[6]により，グラフが連結ならば，辺の集合の，長さ2の鎖への分解が可能であることが示されているくらいで，それ以外にはk鎖を詰める場合も含め，ほとんど論じられていない．

本稿では，木の最大節点パッキング問題と，最大辺パッキング問題，について新たに得た結果[10]について紹介する．詳しくは[10]を参照されたい．

2．諸定義

本節では，以下の議論に必要な定義を与える．なお，よく知られたグラフ理論の用語については，定義せずに用いた場合もある．詳しくは，文献［1, 2, 5］等を参照されたい．

(無向)木とは，閉路を持たない(無向)グラフである．次数(それに接する辺数)が1の節点を葉という．櫛形$(k_1, \ldots, k_h)$木とは長さh－1の道と葉のみとからなっており，道上のi番目の節点(軸)が丁度k_i個の葉を持っている木である．k星とは櫛形(k)木のことである．

根付き有向木T＝(V，E，r)，但し，Vは節点の集合，Eは辺の集合，rは根，は，次の条件を満たす有向木である．

1．根rは親をもたない．

2．根以外のすべての節点は唯一の親をもつ．

3．根以外のすべての節点は根から道により到達できる．

但し，(i，j)∈Eのとき，iはjの親，jはiの子という．子を持たない節点を葉という．また，iからjへ道があるとき，iはjの先祖，jはiの子孫という．なお，そのような道は唯一であることに注意せよ．根付き有向木の高さとは，根から最も遠い葉までの距離をいう．

根付き有向木T＝(V，E，r)の節点vに対し，vを根とし，vの子孫すべてと、Tでそれらを結ぶ辺とからなる部分木を，T(v)と記す．

根付き櫛形$(k_1, \ldots, k_h)$木とは，櫛形$(k_1, \ldots, k_h)$木の最初の軸(k_1個の葉をもつもの)を根として根付き有向木にしたものである．

$$S = \{T_1, \ldots, T_t\},$$

但し，$T_i=(V_i, E_i)$，i＝1，…，t，T_iは(無向または有向木)Tの部分木，がTの節点パッキングとは，任意のi≠j，i＝1，…，t，に対し

$$V_i \cap V_j = \phi$$

が成り立つことであり，SがTの辺パッキングとは，任意のi≠j，i＝1，…，t，に対し

$$E_i \cap E_j = \phi$$

が成り立つことである．以下，本稿では，Sの要素がすべてあるT_0に同型な場合のみを論ずる．最大辺(節点)パッキング問題とは，与えられたTに対して，|S|を最大にする辺(節点)パッキングを求める問題である．但し，|S|は集合Sの要素数を表わす．

3．根付き有向木上の木パッキング

まず，根付き有向木T＝(V，E，r)，rは根，への，根付き有向木T_0の最大節点パッキング問題を解くアルゴリズムＲＴＲＥＥ－Ｎを導く．rから出発して幅優先探索[1]を行ない，訪れた順に節点に番号を付す．以下では，節点と，それに付された番号とを同一視する．次に，番号最大の節点から順に訪れ(つまり，根から遠い方の節点から順に訪れて)，以下の操作を根に到るまで行う．

節点vを訪れたとき，T_0を，vを根とするTの部分木T(v)に詰めることができるかどうか調べるため，Procedure ＣＨＥＣＫ(T_0，T(v))を呼び，ＹＥＳならばT_0をT(v)に詰め，TからT_0を詰めた部分の節点と辺，及び，その部分に接する辺をすべて取り除き，ＮＯならば，そのままにして，次の節点を訪れる．

Procedure CHECK(T_0, T(v))は以下のように実行される．T_0をT(v)に詰めることができるかどうか調べるため，T_0の根r_0の子w_i，i=1，…，d_0，及びvの子v_j，j=1，…，dに対し，CHECK($T_0(w_i)$，T(v_j))を再帰的に呼び，2部グラフG=(N_1，N_2，E)，ここで，N_1={$T_0(w_i)$| i=1，...，d_0}，N_2={T(v_j)| j=1，...，d}，E={($T_0(w_i)$，T(v_j))|$T_0(w_i)$はT(v_j)に詰めることができる}，を作る．GにN_1の節点をすべて覆うマッチングが存在するときYESを，存在しないときNOを出力する．以上のアイデアを形式的に書くと次のアルゴリズムRTREE－NとProcedure CHECK(T，T')を得る．

アルゴリズムRTREE－N

入力：　根付き有向木T=(V，E，r)，及び，Tに詰められる根付き有向木T_0.

出力：　Tに節点パッキングできる根付き有向木T_0の最大個数N.

ステップ1．Tを根rから順に，訪問順に番号を振りながら幅優先探索を行う．N←0，v←n（，n=|V|).

ステップ2．Tの節点を番号の大きい方から順に訪れ，以下の操作を行う．

節点vを訪れたとき，Procedure CHECK(T_0，T(v))を呼ぶ．もし，答がYESなら，N←N+1とし，T_0が詰められた部分のすべての節点と辺，及びその部分に接するすべての辺をTから取り除き，v←v－1とする．さもなければ，v←v－1とする．

ステップ3．もし，vが0ならステップ4へ，さもなければ，ステップ2へ．

ステップ4．停止．□

Procedure CHECK(T，T')

入力：　根付き有向木T=(V，E，r)と根付き有向木T'=(V'，E'，r').

出力：　もし，TをT'に詰めることができるならYES，さもなければ，NO.

Tの根rの子w_1，…，w_d，及び，T'の根r'の子v_1，…，$v_{d'}$に対し，CHECK(T(w_i)，T'(v_j))を再帰的に呼び，2部グラフG=(N_1，N_2，E)，ここで，N_1={T(w_i)| i=1，...，d}，N_2={T'(v_j)| j=1，...，d'}，E={(T(w_i)，T'(v_j))|T(w_i)はT'(v_j)に詰めることができる}，を作る．そして，N_1の節点をすべて覆うマッチングが存在するときYESを，存在しないときNOを出力して，RETURN．□

定理1．アルゴリズムRTREE－Nにより，O(n^2)で，Tに節点パッキングできる根付き有向木T_0の最大個数が求まる．但し，nはTの節点数．

(略証)アルゴリズムの正当性は，アルゴリズムのステップ2の実行に関する帰納法で示せる．また，ステップ2でTの各節点vを1回ずつ訪れ，節点vを訪れた際のProcedure CHECKの再帰的呼出しは，高々vからh以内(hはT_0の高さ)の子孫に及ぶだけであることに注意すると，Tの各節点vでProcedure CHECK(T_0，T(v))の実行に伴い，解かなければならない最大マッチング問題の回数は高々h回であり，最大マッチング問題を解くのに要する時間がO(|N_1|^2|N_2|)である[8]ことと併せて，全体の計算時間がO(n^2)であることが分る．但し，T_0の節点数は定数とみなすので，|N_1|及びhは定数であることに注意せよ．□

アルゴリズムＲＴＲＥＥ－Ｎでは，ステップ２で節点ｖを訪れたときには，T_0がＴ(ｖ)に詰められるかどうか調べるだけでよかった．それに対し，T_0のＴへの最大辺パッキングを求める場合には，次の２点で難しくなる．

(1)Ｔ(ｖ)にT_0をなるたけ多く詰めなければならない．

(2)そのような最大個数の辺パッキングのうち，後でｖの先祖を訪れたときに"有利"となる可能性のあるものはすべて残す必要がある．

まず，(1)を解決するため，次の問題Ａを考える．

問題Ａ．アルゴリズムＡの直前で定義された２部グラフ$G=(N_1, N_2, E)$が与えられたとき，次の２つの条件を満たす最大個数の辺の集合$M_1, \ldots, M_k$を求めよ．

条件１．M_i，$i=1, \ldots k$，はそれぞれ，N_1の節点をすべて被覆するマッチングである．

条件２．N_2の各節点は，$M_1 \cup \ldots \cup M_k$の辺により，高々一回しか被覆されない．□

問題Ａは次の問題Ａ'が解ければｋに関して２分探索($0 \leqq k \leqq D$，$D(\leqq |N_2|)$は，N_1に属する節点のＧでの最大次数)することで解くことができる．

問題Ａ'．２部グラフ$G=(N_1, N_2, E)$が与えられたとき，問題Ａのふたつの条件を満たす辺の集合$M_1, \ldots, M_k$が存在するかどうか決定せよ．□

問題Ａ'は容易に分るように，次の問題Ｂと等価である．

問題Ｂ．２部グラフ$G=(N_1, N_2, E)$に対して，N_1の節点を全て含み、それらの次数がすべてｋであり，N_2に属する節点の次数がすべて１であるＧの部分グラフG_0(次数制限マッチング)が存在するか？なお、G_0はN_2の節点を必ずしもすべて含む必要はない．□

よく知られているように[2]，問題Ｂは容易に，Ｇに対し、節点ｓ，ｔ，及び，各$x \in N_1$に対し，辺(ｓ，ｘ)を，また，各$y \in N_2$に対し，辺(ｙ，ｔ)を付加し，更に各$x \in N_1$に対し，$c(s, x)=k$，各$x \in N_1$，$y \in N_2$に対し，$c(x, y)=\infty$，また，各$y \in N_2$に対し，$c(y, t)=1$，但し、$c(i, j)$は辺(ｉ，ｊ)の容量，として得られるネットワークＳ上での最大フロー問題に帰着することができ，たとえば，SleatorとTarjanの方法[11](Ｏ(ｍｎｌｏｇ ｎ)，但し，ｎ，ｍは，それぞれネットワークの節点数と辺数)を用いると，$O(|N_1||N_2|(|N_1|+|N_2|)\log(|N_1|+|N_2|))$時間で解くことができる．

次に，上記の(2)を解決するため，動的計画法を用いる．すなわち，ステップ２で節点ｖを訪れたとき，ｖの子v_i，$i=1, \ldots, d$，に対して残されているv_iを根とする各部分木$T(v_i, j)$からv_i，$i=1, \ldots, d$に対してひとつずつ選んで得られる各組合せそれぞれに対して最大個数の辺パッキングを求め，全体での最大k_Mを求める．ところが，節点ｖの子の数をｄとすると，k_Mを求めるためには，c^d個のｖの子を根とする部分木の組合せを考えなければならず，そのままでは，計算時間を多項式時間に押えるためには，Ｔの最大次数を定数としなければならない．そこで，問題Ｂの代りに，次の問題Ｂ'を考える．

問題Ｂ'．$N'_1=\{T_0(w_i) \mid w_i$はT_0の根r_0の子$\}$，$N'_2=\{T(v_i, j) \mid v_i$はｖの子$\}$，$E'=\{(T_0(w_i), T(v_p, j)) \mid T_0(w_i)$は$T(v_p, j)$に詰めることができる$\}$として，２部グラフ$G''=(N'_1, N'_2, E')$を作り，更に，節点ｓ，ｔ，１，２，...，ｄを追加し，また，各$x \in N'_1$に対し，辺(ｓ，ｘ)，$j=1, \ldots, d$に対し，辺(ｊ，ｔ)，ｖの子v_jを根とする部分木に対応する各$y \in N'_2$に対し，辺(ｙ，ｊ)，また，各辺の容量を，各$x \in N'_1$に対

し，c(s，x)＝k，各x∈N'$_1$，y∈ N'$_2$に対し，c(x，y)＝∞，各y∈N'$_2$に対し，c(y，j)＝∞，j＝1，...，dに対し，c(j，t)＝1，として得られる，ネットワークS'が実行可能流れを持つか.□

問題B'は，最大フロー問題を解くことによって，解くことができる．その解をk_Mとする．次に、T_0の各節点uに対して，次の操作を行う．uの子u_1，...，$u_{t(u)}$，t(u)はuの子の数，に対してvの各子v_i，i＝1，...，d，に対応する部分木を丁度一個ずつ，計t(u)個だけ選び、それらからなる部分木に$T_0(u)$が詰められるかどうか調べる．詰められるとき、それら計t(u)個を除いた残りのvの子を根とする部分木をひとつずつ選んだすべての組合せに対してk_M個のパッキングが可能か問題B'と同様にして調べ，詰められる組合せが存在する場合はひと組だけ残してT_0をk_M個詰め，その部分の辺を取り除く．そのようにして得られた部分木のうちで重複を省き，他に含まれないT(v，j)のみ残せばよい．

以上のアイデアを形式的に書くと，次のアルゴリズムＲＴＲＥＥ－Ｅを得る．

アルゴリズムＲＴＲＥＥ－Ｅ

入力：　根付き有向木Ｔ＝(Ｖ，Ｅ，ｒ)，及び，Ｔに詰められる根付き有向木T_0.

出力：　Ｔに辺パッキングできる根付き有向木T_0の最大個数Ｎ.

ステップ１．Ｔを根ｒから順に，訪問順に番号を振りながら幅優先探索を行う．Ｎ←０，ｖ←ｎ（，ｎ＝|Ｖ|).

ステップ２．Ｔの節点を番号の大きい方から順に訪れ，以下の操作を行う．

節点ｖを訪れたとき，ｖの子v_i，ｉ＝１，...，ｄに対して残されているv_iを根とする部分木Ｔ(v_i，ｊ)に対して，問題Ｂ'をｋに関する２分探索を行ないながら解くことにより，ｋの最大値k_Mを求める．次に問題Ｂ'の直後に述べたようにして部分木Ｔ(ｖ，ｊ)を生成し，Ｎ←Ｎ＋k_M，ｖ←ｖ－１とする．

ステップ３．もし，ｖが０ならステップ４へ，さもなければ，ステップ２へ．

ステップ４．停止．□

定理２．アルゴリズムＲＴＲＥＥ－Ｅにより，根付き有向木T_0の最大辺パッキングが多項式時間で求まる．□

もし，T_0が根付き櫛形(k_1，...，k_h)木(ｈは定数)なら，アルゴリズムＲＴＲＥＥ－Ｎのステップ２において次のProcedure ＣＯＭＢＣＨＥＣＫ(k_1，...，k_h，Ｔ(ｖ))を用いることによって，計算の複雑度をＯ(ｎ)に下げることができる．このようにして得られるアルゴリズムをＲＣＯＭＢ－Ｎと呼ぶ．

Procedure ＣＯＭＢＣＨＥＣＫ(k_1，...，k_h，Ｔ(ｖ))

ｖの子をv_1，...，v_dとする．ｉ＝１，...，ｄに対してProcedure ＣＯＭＢＣＨＥＣＫ(k_2，...，k_h，Ｔ(v_i))を再帰的に呼ぶ．少なくともひとつＹＥＳ，かつ，$k_1 \leqq d-1$ならＹＥＳ，さもなければ，ＮＯ．□

定理３．アルゴリズムＲＣＯＭＢ－Ｎにより，Ｏ(ｎ)で根付き櫛形$(k_1, \ldots, k_h)$木の最大節点パッキングが求まる．□

最大辺パッキング問題の場合も，根付き櫛形$(k_1, \ldots, k_h)$木を詰める場合は，以下のように計算複雑度を下げることが可能である．

アルゴリズムＲＣＯＭＢ－Ｅ

入力：　根付き有向木Ｔ＝(Ｖ，Ｅ，ｒ)，及び，Ｔに詰められる根付き櫛形$(k_1, \ldots, k_h)$木T_0．

出力：　Ｔに辺パッキングできるT_0の最大個数Ｎ．

ステップ１．Ｔを根ｒから順に，訪問順に番号を振りながら幅優先探索を行う．Ｎ←０，ｖ←ｎ（，ｎ＝|Ｖ|）．

ステップ２．Ｔの節点を番号の大きい方から順に訪れ，以下の操作を行う．

節点ｖを訪れたとき，n_1を，ｖの子v_iのうち，それを根とする部分木Ｔ(v_i, j)で，根付き櫛形$(k_2, \ldots, k_h)$木を詰めることのできるものがあるものの個数，n_2を，そうでないｖの子の個数とする．すると，

$n_1 \leq \lfloor n_2 / k_1 \rfloor$ならば$n_1$個の根付き櫛形$(k_1, \ldots, k_h)$木を詰める．さもなければ，$\lfloor n_2 / k_1 \rfloor + ((n_2 - k_1 \lfloor n_2 / k_1 \rfloor) + (n_1 - \lfloor n_2 / k_1 \rfloor)) / (k_1 + 1)$個の根付き櫛形$(k_1, \ldots, k_h)$木を詰める．そのようなｖにおける最大個数の詰め方から，根付き櫛形$(k_{h-i+1}, \ldots, k_h)$木が詰められる部分木Ｔ(v_i, i)があればそれぞれひとつずつ残す．

ステップ３．もし，ｖが０ならステップ４へ，さもなければ，ステップ２へ．

ステップ４．停止．□

定理４．アルゴリズムＲＣＯＭＢ－Ｅにより，Ｏ(ｎ)で根付き櫛形$(k_1, \ldots, k_h)$木の最大辺パッキングが求まる．□

４．無向木への無向木のパッキング

本節では，まず，無向木Ｔへの無向木T_0の最大節点パッキングを求めるアルゴリズムＴＲＥＥ－Ｎを導入する．ＴＲＥＥ－Ｎでは，まず，Ｔの任意の節点ｒを根として選び，ｒから訪問順に番号を振りながら幅優先探索を行う．以下では得られた根付き有向木もＴで表わす．次に，Ｔを，番号の大きな節点から順に訪れ，以下の操作を行う．節点ｖを訪れたときを考える．根付き有向木上のパッキングの場合には，T_0の根が指定されていたが，無向木の場合には，T_0のどの節点を根としてＴに詰めてもよい．そこで，T_0の各節点ｕに対し，ｕを根とする根付き有向木T_0^uを作り，ＣＨＥＣＫ$(T_0^u, T(v))$をそれぞれ実行する．ひとつでもＹＥＳなら，詰められる．詰められる場合には，T_0を詰めた部分の節点，辺，及び，それに隣接する全ての辺をＴから除去して，すべてＮＯならそのまま，いずれも，次の節点を訪れる．

以上のアイデアを形式的に書くと，次のアルゴリズムＴＲＥＥ－Ｎを得る．

アルゴリズムＴＲＥＥ－Ｎ

入力：　無向木Ｔ＝(Ｖ，Ｅ)，及び，Ｔに詰められる無向木T_0．

出力： Tに節点パッキングできる無向木T_0の最大個数N．

ステップ１．Tの節点rを任意に選び，rから順に，訪問順に番号を振りながら幅優先探索を行い，得られた根付き有向木もTと呼ぶ．rを根とする根付き有向木をつくる．N←0，v←n(，n=|V|)．

ステップ２．T_0の各節点uに対してProcedure CHECK(T_0^u，T(v))を呼ぶ．もし，少なくともひとつ答がYESなら，N←N＋１とし，T_0を詰めた部分のすべての節点と辺，及びその部分に接するすべての辺をTから取り除き，v←v－１とする．さもなければ，v←v－１とする．

ステップ３．もし，vが0ならステップ４へ，さもなければ，ステップ２へ．

ステップ４．停止．□

定理１と同様にして，次の定理を得る．

定理５．アルゴリズムTREE－Nにより，$O(n^2)$でTに詰められる無向木T_0の最大個数が求まる．□

次に，無向木Tへの無向木T_0の最大辺パッキングを求める多項式時間のアルゴリズムTREE－Eを導く．

ステップ２で節点vを訪れたとき，根付き有向木の場合と異なり，T_0のどの節点を根として詰めるかという自由度がある．そこで，RTREE－Eで述べた問題Aの代りに，次の問題A_0を考える．

問題A_0．問題Aの２部グラフG＝(N_1，N_2，E)に対して，$N_{1u}=\{T_0^u(w_i)\mid w_i$はuの子$\}$，u＝1，...，$n_0$、但し、$n_0$は$T_0$の節点数、を作り，その和集合を$N'_1$とする．$T_0^u(w_i)\in N'_1$が$x\in N_2$に詰められるときかつそのときに限り，xと辺で結ぶ，として得られるグラフを$G'=(N'_1, N_2, E')$とする．そのとき，次の条件を満たす辺の集合M_{ui}，i＝１，...，k_u，を最大個数とる．

条件１．M_{ui}はN_{1u}，かつ，それのみに属する節点全てを被覆するマッチングである．

条件２．N_2の各節点は，$\cup M_{ui}$の辺により，高々一回しか被覆できない．□

問題A_0は次の問題A_0'が解ければkに関して２分探索(0≦k≦D，Dは，N_1に属する節点の最大次数)することで解くことができる．ただし，$k=k_1+\ldots+k_{n0}$を満たす組k_1，...，k_{n0}すべてについて解かなければならない．

問題A_0'．２部グラフ$G'=(N'_1, N_2, E)$が与えられたとき，$k=k_1+\ldots+k_{n0}$を満たす組k_1，...，k_{n0}について問題A_0の２つの条件を満たす辺の集合M_{u1}，...，M_{uku}が存在するかどうか決定せよ．□

問題A_0'は容易に分るように，次の問題B_0と等価であり，また，問題B_0は，問題Bと同様にして最大フロー問題に帰着して解ける．

問題B_0．２部グラフ$G'=(N'_1, N_2, E)$に対して，N_{1u}，u＝1，...，n_0の節点を全て含み、それらの次数がすべてk_uであり，N_2に属する節点の次数がすべて１である次数制限マッチングを持つか？□

更に，問題B'と同様にして，計算時間を多項式に保つため，次の問題B'_0を考える．

問題B'_0. $N'_2=\{T(v_p, j) \mid v_p$はvの子$\}$, $E'=\{(T_0(w_i), T(v_p, j)) \mid T_0(w_i)$は$T(v_p, j)$に詰めることができる$\}$として，２部グラフ$G''=(N'_1, N'_2, E')$を作り，更に，節点ｓ，ｔ，１，２，...，ｄを追加し，また，各$x \in N'_1$に対し，辺(ｓ，ｘ)，ｊ＝1,...,dに対し，(ｊ，ｔ)，ｖの子v_jを根とする部分木に対応する各$y \in N'_2$に対し，(ｙ，ｊ)，また，各辺の容量を，各$x \in N'_1$に対し，ｃ(ｓ，ｘ)＝ｋ，各$x \in N'_1$，$y \in N'_2$に対し，ｃ(ｖ，ｙ)＝∞，各$y \in N'_2$に対し，ｃ(ｙ，ｊ)＝∞，ｊ＝１，...，ｄに対し，ｃ(ｊ，ｔ)＝１，として，ネットワークＳ'が実行可能流れを持つか.□

問題B'_0は，最大フロー問題を解くことによって，解ける．その解をk_Mとする．あとは，問題Ｂ'の直後に述べたようにして部分木Ｔ(ｖ，ｊ)を生成する．

アルゴリズムＴＲＥＥ－Ｅ

入力：　無向木Ｔ＝(Ｖ，Ｅ)，及び，Ｔに詰められる無向木T_0.

出力：　Ｔに辺パッキングできる無向木T_0の最大個数.

ステップ１．Ｔの節点ｒを任意に選び，ｒから順に，訪問順に番号を振りながら幅優先探索を行う．Ｎ←０，ｖ←ｎ（，ｎ＝|Ｖ|）．得られた根付き有向木もＴと呼ぶ.

ステップ２．Ｔの節点を番号の大きい方から順に訪れ，以下の操作を行う.

節点ｖを訪れたとき，ｖの子w_i，ｉ＝１，...，ｄに対して残されているw_iを根とする部分木Ｔ(w_i，ｊ)に対して，問題B'_0をｋに関する２分探索を行ないながら解くことにより，ｋの最大値k_Mを求める．あとは，問題Ｂ'の直後に述べたようにして部分木Ｔ(ｖ，ｊ)を生成する．ｖ←ｖ－１とする.

ステップ３．もし，ｖが０ならステップ４へ，さもなければ，ステップ２へ.

ステップ４．停止．□

定理６．アルゴリズムＴＲＥＥ－Ｅにより，無向木Ｔに対する無向木Ｔ'の最大辺パッキングが多項式時間で求まる．□

次に，無向木に，特にｋ星を詰める場合，節点パッキング，辺パッキングのいずれもＯ(ｎ)で解けることを示す.

まず，ｋ星の最大節点パッキングをＯ(ｎ)で求めるアルゴリズムｋＳＴＡＲ－Ｎから述べる．やはり，まず，Ｔの任意の節点ｒを根として選び，ｒから順に番号を振りながら幅優先探索を行い，根付き有向木とする(それもＴと呼ぶ)．次に，Ｔの節点を番号の大きな方から順に訪れ，以下の操作を行う.

節点ｖを訪れたとき，ｋ星を詰めることができるかどうか調べる．そのとき，Ｔ(ｖ)へのｋ星の詰め方には，以下の２通りがあることに注意せよ.

型１．葉がｖに重なる詰め方.

型２．軸がｖに重なる詰め方.

ｋ星の型１かまたは型２のいずれかの詰め方が可能かどうか調べる．可能なら詰め，詰めた部分の節点及び辺，及び，その部分に隣接する辺をすべてＴから除去し，次の節点を訪れる．紙数の都合で詳細は略す.

次に，k星の最大辺パッキングをO(n)で求めるアルゴリズムkSTAR－Eも以下のようにして得るが，紙数の都合で詳細は略す．ステップ2で節点vを訪れたとき，まず，型1の詰め方をできるだけ多く行う．次に，残りの部分木に，型2の詰め方をなるたけ多く行う．

本節を閉じるにあたり，櫛形(k_1, k_2)木の無向木への節点及び辺パッキングが，いずれもO(n)で解けることを示そう．

まず，節点パッキングについては，ステップ2で，節点vを訪れたときvの子，$w_1, \ldots, w_d$のうち，k_1個以上の子を持つものがありかつ$d-1 \geqq k_2$，または，k_2個以上の子を持つものがありかつ$d-1 \geqq k_1$，なら詰める．詳細は略す．

次に，櫛形(k_1, k_2)木の無向木Tへの最大辺パッキングをO(n)で求めるアルゴリズム2COMB－Eでは，節点vを訪れた際，なるたけ多く，根付き櫛形(k_1, k_2)木，又は，根付き櫛形(k_2, k_1)木を詰めるが，そのとき，$k_1 \geqq k_2$とすると，まず，根付き櫛形(k_2, k_1)木を優先し，次に根付き櫛形(k_1, k_2)木を詰める．詳細は略す．

5．むすび

今後の課題として，グラフパッキング問題で，今までに得た以外の効率良く解ける部分クラスを求め，多項式時間で解ける限界を明らかにすること，良い近似解法を開発し，その性能評価を行うこと，などがある．

謝辞　末筆ではあるが，日頃ご指導賜わる京都大学工学部の茨木俊秀教授と長谷川利治教授，及び，関西大学工学部の三根久教授(京都大学名誉教授)に深謝する．また，第3節の問題Aが問題Bに帰着できることを示唆していただいた，京都大学工学部の永持仁氏に謝意を表する．なお，本研究は一部文部省科学研究費によるものである．

文　献

[1] Aho, A. V., Hopcroft, J. E. and Ullman, J. D., The Design and Analysis of Computer Algorithms, Addison-Wesley(1974)(邦訳，野崎，野下，共訳，アルゴリズムの設計と解析I,II，サイエンス社(昭和52)).

[2] Berge, C., Graphes et Hypergraphes, Dunod (1970)；C. ベルジュ著，伊理他訳，グラフの理論I，サイエンス社(1976).

[3] Boesch, F. T. and Gimpel, J. F., "Covering the points of a digraph with point-disjoint paths and its application to code optimization", J. Assoc. Comput., Mach., 24, pp.192-198(1977).

[4] Garey, M. R. and Johnson, D. S., Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H., Freeman and Company, San Fransisco, (1979).

[5] Harary, F., Graph Theory, Addison-Wesley, Reading Mass.；池田貞雄(訳)，グラフ理論，共立出版(昭和46).

[6] Jünger, M., Reinelt, G. and Pulleyblank, W. R., "On partitioning the edges of graphs into connected subgraphs", J. Graph Theory, Vol.9, pp.539-549 (1985).

[7] Kirkpatrick, D. G. and Hell, P., "On the complexity of general graph factor problems", SIAM J. Comput. Vol.12, No.3, pp.601-609(1983).

[8] Lawler, E. L., Combinatorial Optimization: Networks and Matroids, Holt Rinehalt and Winston(1976).

[9] Masuyama, S. and Ibaraki, T., "Computational complexity of chain packing problems", Technical Report #87001, Department of Applied Mathematics & Physics, Faculty of Engineering, Kyoto University, Kyoto 606, Japan, (1987).

[10] Masuyama, S., "On the tree packing problems", manuscript under preparation for the Technical Report of Department of Applied Mathematics & Physics, Faculty of Engineering, Kyoto University, Kyoto 606, Japan, (1987).

[11] Sleater, D. D. and Tarjan,R. E., "A data structures for dynamic trees", Proc. of the 13th Annual ACM Symposium on Theory of Computing, pp.114-122 (1981).

[12] Zhang, Ze-Zeng, Masuyama, S., Ibaraki, T. and Mine, H., "Graph packing over a rooted tree", to appear in Memoirs of the Faculty of Engineering, Kyoto University, Vol.49, No.2, Kyoto, Japan.